Find: A Loan for Me   Powered by GetSmart.com

Intellectual Property Network

Home | Search | Order | Shopping Cart | Login | Site Map | Help

WO101206A2: SYSTEM DYNAMICS MODEL BUILDER AND SIMULATOR

No Image | View Cart | View INPADOC only

Add to cart: More choices...

Country:	**WO** World Intellectual Property Organization (WIPO)
Kind:	**A2** Publ.OF the Int.Appl. without Int.Search REP.
Inventor(s):	**EUBANKS, C., Keith**, 119 Oakland Avenue, Arlington, MA 02174, United States of America
	GRAHAM, Alan, K., 37 Stoneymeade Way, Acton, MA 01720, United States of America
	YEAGER, Larry, 261 Main Street, Bolton, MA 01740, United States of America
	EVANS, Andy, 45 Drum Hill Road, Concord, MA 02174, United States of America
	DIEGUEZ, Gregg, Tudor Glen Village, Unit 24-C-1, 111 Locust Street, Woburn, MA 01801, United States of America
	FINE, John, S., 45 Meadowbrook Road, Bedford, MA 01730, United States of America
	CANOVI, Adolfo, 175 Beacon Street #405, Somerville, MA 02143, United States of America
	HEYE, Chris, 101 Dean Street, Belmont, MA 02478, United States of America
	MAXIMOV, Peter, 2 Glenbrook Lane #17, Arlington, MA 02474, United States of America
	CHAN, Harry, 77 Robertson Street, Quincy, MA 02169, United States of America
	VON LAUDERMAN, Karl, 285 Massachusetts Avenue, Arlington, MA 02474, United States of America
Applicant(s):	**STRATEGIC SIMULATION SYSTEMS, INC.**, 41 William Linskey Way, Cambridge, MA 02142, United States of America
	News, Profiles, Stocks and More about this company
Issued/Filed Dates:	**Jan. 4, 2001** / June 28, 2000
Application Number:	**WO2000US0017793**
IPC Class:	<u>G05B 13/00;</u>
Priority Number(s):	July 23, 1999 **US1999009360044**

Legal Status:

| <u>Gazette date</u>	Code	Description (remarks) List **all possible codes for WO**
Jan. 4, 2001	**A2**	Publication of the international application without the international search report
Jan. 4, 2001	**AL**	Designated countries for regional patents (AT BE CH CY DE DK ES FI FR GB GR IE IT LU MC NL PT SE)
June 28, 2000	**AE**	International application
July 23, 1999	**AA**	Priority claimed

Designated Countries: **European patent:** AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL,

PT, SE

Abstract: A system dynamics modeling environment has a graphical user interface to allow a model developer to define a model through graphical inputs, for example by selecting icons and dragging them onto a representation of a flow diagram. The data and other information regarding a particular model are stored within a database structure that efficiently organizes the data. A second non-graphical editor is provided as a separate and distinct tool for editing the equations and other information representative of a model. The system dynamics modeling environment allows the definition of a plurality of groups, with each group consisting of a data set representing a complete flow diagram and capable of being coupled to another group defining a portion of the model on the same layer of the model. An efficient simulation environment is provided that determines which groups within a model are active and performs simulations only on those active groups. Users will also have the ability to create customized user interfaces, or "dashboards" for their models, without requiring programming experience. Users also have access to features that make it easy to create and save scenarios and models.
[Show "fr" Abstract]

Representative Image:

[x] Main
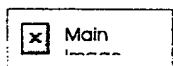
[Show "fr" image]

Attorney, Agent, or Firm: **WRIGHT, William, H.;**

Other Abstract Info: none

Foreign References: (No patents reference this one)

Powered by DB2 and Net.Data

Nominate this invention for the Gallery...

Alternative Searches   Patent Number   Boolean Text   Advanced Text

Browse   U.S. Class by title   U.S. Class by number   IBM Technical Disclosure Bulletin

Privacy | Legal | Gallery | IP Pages | Advertising | FAQ | Contact Us

▲ **Presentation:** `Basic` ▓ **Image:** `Small` ▓   Français ▮▮
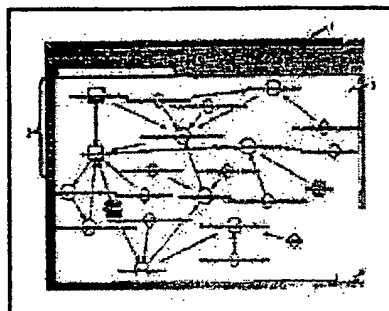1 of 1

PUBLISHED INTERNATIONAL APPLICATION

(11) **WO 01/01206**   (13)   A2

(21) PCT/US00/17793

(22) **28 June 2000 (28.06.2000)**

(25) ENG   (26)   ENG

(31) 60/142,029   (32)   **30 June 1999**   (33)   US
                                     (30.06.1999)

(31) 09/360,044   (32)   **23 July 1999**   (33)   US
                                     (23.07.1999)

(43) 04 January 2001 (04.01.2001)

$(51)^7$ G05B 13/00

(54) SYSTEM DYNAMICS MODEL BUILDER AND SIMULATOR

(71) **STRATEGIC SIMULATION SYSTEMS, INC.** 41 William Linskey Way, Cambridge, MA
      02142; (US). [US/US].

(72) **EUBANKS, C., Keith** 119 Oakland Avenue, Arlington, MA 02174; (US). **GRAHAM, Alan,
      K.** 37 Stoneymeade Way, Acton, MA 01720; (US). **YEAGER, Larry** 261 Main Street,
      Bolton, MA 01740; (US). **EVANS, Andy** 45 Drum Hill Road, Concord, MA 02174; (US).
      DIEGUEZ, Gregg Tudor Glen Village, Unit 24-C-1, 111 Locust Street, Woburn, MA 01801;
      (US). **FINE, John, S.** 45 Meadowbrook Road, Bedford, MA 01730; (US). **CANOVI, Adolfo**
      175 Beacon Street #405, Somerville, MA 02143; (US). **HEYE, Chris** 101 Dean Street,
      Belmont, MA 02478; (US). **MAXIMOV, Peter** 2 Glenbrook Lane #17, Arlington, MA
      02474; (US). **CHAN, Harry** 77 Robertson Street, Quincy, MA 02169; (US). **VON
      LAUDERMAN, Karl** 285 Massachusetts Avenue, Arlington, MA 02474; (US).

(74) **WRIGHT, William, H.** Hogan & Hartson LLP, Biltmore Tower, 500 South Grand Avenue,
      Suite 1900, Los Angeles, CA 90071; (US).

(81) EP (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE)

Abstract

A system dynamics modeling environment has a graphical
user interface to allow a model developer to define a
model through graphical inputs, for example by selecting
icons and dragging them onto a representation of a flow
diagram. The data and other information regarding a
particular model are stored within a database structure that
efficiently organizes the data. A second non-graphical
editor is provided as a separate and distinct tool for editing
the equations and other information representative of a
model. The system dynamics modeling environment
allows the definition of a plurality of groups, with each group consisting of a data set representing a
complete flow diagram and capable of being coupled to another group defining a portion of the
model on the same layer of the model. An efficient simulation environment is provided that
determines which groups within a model are active and performs simulations only on those active
groups. Users will also have the ability to create customized user interfaces, or "dashboards" for their
models, without requiring programming experience. Users also have access to features that make it

easy to create and save scenarios and models.

(54) Title: SYSTEM DYNAMICS MODEL BUILDER AND SIMULATOR

(57) Abstract: A system dynamics modeling environment has a graphical user interface to allow a model developer to define a model through graphical inputs, for example by selecting icons and dragging them onto a representation of a flow diagram. The data and other information regarding a particular model are stored within a database structure that efficiently organizes the data. A second non-graphical editor is provided as a separate and distinct tool for editing the equations and other information representative of a model. The system dynamics modeling environment allows the definition of a plurality of groups, with each group consisting of a data set representing a complete flow diagram and capable of being coupled to another group defining a portion of the model on the same layer of the model. An efficient simulation environment is provided that determines which groups within a model are active and performs simulations only on those active groups. Users will also have the ability to create customized user interfaces, or "dashboards" for their models, without requiring programming experience. Users also have access to features that make it easy to create and save scenarios and models.

5   SYSTEM DYNAMICS MODEL BUILDER AND SIMULATOR

COPYRIGHT NOTICE:

A portion of the disclosure, including the drawings, of this patent document

contains material subject to copyright protection. The copyright owner has no objection

10   to the facsimile reproduction by anyone of the patent document or the patent disclosure,

as it appears in the Patent and Trademark Office patent file or records, but otherwise

reserves all copyright rights whatsoever.

PRIORITY:

15   This application claims priority from U.S. provisional patent application Serial

No. _____, filed June 30, 1999, which application is hereby incorporated by

reference in its entirety.

BACKGROUND OF THE INVENTION

20   1.   Field of the Invention

The present invention relates to systems, methods and products for developing

and executing system dynamics models.

2.   Description of the Related Art

System dynamics and system dynamics modeling provide tools for analyzing

complex systems. For example, system dynamics modeling has been applied to

biological systems including the production and regulation of insulin within a human

body, to educational processes including modeling how students learn, to social systems

5   including modeling the relationship between crime and drug availability, and to a variety

of economic and business situations. By applying the principals of system dynamics,

computational models with qualitative and, often, quantitative accuracy can be developed

and can be used to gain an understanding of the modeled system. Even when a model is

highly schematic, insights can be drawn simply because the system dynamics model

10   adequately represents the structure and relationships among elements of a system. Causal

relationships that are not evident in a static model can become apparent in a system

dynamics model.

Once an accurate system dynamics model is developed, the model can be used to

perform computer-based experiments and simulations to evaluate changes in structure

15   and in the coupling between structures, and how varying the initial conditions or input

variables affect the system. For example, control mechanisms such as rules or policies

can be tested on a system dynamics model to determine if the control will have the

desired effect and whether the control is likely to be cost effective or advantageous.

What has allowed system dynamics modeling to be applied to such diverse

20   problems is the generality of the system dynamics methodology. In developing a system

dynamics model, a model developer breaks down the system or problem to be analyzed

into constituent elements and defines the relationships among the various elements of the

system or problem. To the extent possible, the model developer attempts to define a

2

model in terms of fundamental constituent elements so that the model is as simple as

possible while still having an appropriate level of structural detail. One way in which

fundamental elements can be characterized is by identifying the minimum set of variables

that describe the system. The relationships among the fundamental elements are

5   translated into the relationships among the variables that describe the system.

Relationships between any two elements can be defined functionally, that is in the form

of an equation, or empirically, for example using a table listing an observed set of

relationships.

       Within the system dynamics literature, variables that describe the state of the

10   system are sometimes referred to as "stocks" or "levels" and the relationships between

levels are defined in terms of "flows" or "rates." Another concept typically included in

system dynamics models is that of an "auxiliary," which receives input data and converts

or manipulates that input data to provide output data. "Connectors" are often used to

illustrate graphically the flow of information among stocks, flows and auxiliaries.

15   One fundamental aspect of system dynamics is the explicit consideration of time

in the model. Most all of the relationships among variables are time varying, which is a

characteristic of system dynamics modeling that allows its models to represent real world

systems. Simulating a system dynamics model causes the model to advance through a

series of time steps, with many of the variables changing with each time step. Because

20   time is explicitly considered along with the relationships among different variables,

feedback and environmental effects are an important aspect of system dynamics models.

Using a system dynamics model allows the impact of changes in a variable to be

observed, whether to evaluate the accuracy of a model or to understand the impact that

3

changes in the system variables have on the system as a whole. Graphs of system variables allow aspects of the system to be monitored, illustrating both transient behavior and steady state behavior, if it exists.

System dynamics models present a multivariable view of a system, focusing both on the constituent elements of the system and the interaction among those elements. Variables interact with one another due to a variety of feedback mechanisms, causing the values of the variables, and thus the response of the system as a whole, to change over time. Instead of focusing on the independent activity of isolated sub-units, system dynamics attempts to model the structure of the entire system, the interaction among variables in the structure and the way that this interaction changes the variables over time. In this way, the behavior of complex systems can be modeled as a function of time to provide a tool for understanding the system. Further, analysis of structural changes in part of the system can be observed to determine their effect on the system as a whole.

Conventional model optimization is proscriptive, in that an initial goal is set and the model is tested under various conditions. The set of conditions that comes closest to the predetermined goal is deemed to be the "optimal solution." On the other hand, system dynamics simulations tend to be more descriptive, in that the initial conditions are set and the computer determines what would happen to the system over a particular time interval. System dynamics simulations are thus far more often "what-if" models. This does not necessarily preclude system dynamics models from being used for forecasting discrete events. Developing a validated, predictive system dynamics model capable of such prediction requires a significant level of testing, observations and adjustments to

4

reach accuracy. Once such a model is developed, though, it can be used to predict system response with good reliability.

A system dynamics model typically represents a set of linear and non-linear relationships among variables, with many of the relationships mathematically defined as
5   differential equations or difference equations. These equations are integrated over time to generate a simulation. Stepping or cycling the model through time conventionally requires that all of the equations that make up the model be approximated or solved in each time interval or step. The complexity of these models requires the use of computers to fully understand the effect of changes to all but the very simplest of systems.
10   Consequently, system dynamics models are implemented in computer systems, with complex models testing the limits of conventionally available computer systems.

Construction and simulation of system dynamics models involves: (i) developing a model structure that is representative of the physical system relevant to the problem, including defining the variables that are representative of the system, (ii) defining the
15   behavior of variables in the system, which are essentially a set of decision-making rules, (iii) setting initial conditions for the variables of the model, (iv) running the simulation for a specified time period or number of cycles to determine the overall change in behavior of the system, (v) testing the model by comparing it to observations, and (vi) modifying the model, as required. Tests of the model determine its validity or relevance
20   and hence its usefulness. Tests might include model causal tests and model behavior tests. For instance, if a simulation of a population model shows a negative living population level when run for a particular time period, the model needs to be adjusted because it produced an outcome that is not physically possible.

5

In developing a system dynamics model, the boundaries of the system to be

modeled are established to encompass the identified problem. Defining the boundaries of

the problem is significant, in that the model developer selects structures and functions to

include as being determinative of system behavior and selects other structures and

5   functions not to include because they are not sufficiently important. The structures and

functions to be modeled are associated with variables and relationships. Because system

dynamics models tend to be computationally intensive, it is preferable to tailor the model

precisely while still leaving sufficient detail to accurately represent the system. The

result of this model conceptualization stage may be a simplified flow or causal loop

10   diagram representing the variables of a system with their interrelationships.

Next, the diagram is translated into a computer useable form. This is generally

accomplished with a program specifically developed for system dynamics modeling.

When model definition produces a flow diagram, translating the model into a computer

executable model is most easily accomplished through a graphical user interface. There

15   are presently a few software products for system dynamics model construction and

simulation on the market that provide a graphical user interface in which a flow diagram

can be entered graphically in a workspace so that the flow diagram is reproduced on the

screen of a computer. For instance, "STELLA" (System Thinking Educational Learning

Lab with Animation) available from High Performance Systems, Inc., Powersim

20   "Constructor" from Powersim, and "VENSIM" from Ventana Software are examples of

icon-driven system dynamics modeling tools. These programs allow a model developer

to define a system dynamics model in terms of a flow diagram entered into the computer

graphically. STELLA, and to a lesser extent, Powersim Constructor, are designed

primarily to support the construction and simulation of relatively small system dynamics models that can be simulated in a reasonable amount of time. As such, they tend to be ill-suited to storing and executing large models, *i.e.*, those with thousands, or tens of thousands, of equations and array elements.

5   Another product, "DYNAMO" is capable of running large models, but lacks the graphical, model-building capability and other features related to ease of use. Moreover, while this program has advantages for large model definition, the program's simulation engine is based on decades-old programming technologies and is therefore slow as compared to, for example, Vensim, as a platform for running large models. In addition, a

10   user must have a knowledge of Visual Basic, C++ or a similar programming language to develop a professional-looking interface with DYNAMO. This restriction limits a user's ability to design and create decision support and other software applications that can be used by third parties who do not possess a detailed understanding of how the simulation software itself functions. This is true of Powersim, Vensim, and STELLA as well. While

15   these programs do provide rudimentary interface building capabilities, users without programming capabilities cannot create professional, packaged applications with these software programs.

There is a growing interest in the use of system dynamics modeling to model business situations and to inform decision making in complex environments. Tools are

20   needed to facilitate the creation and rapid simulation of large system dynamics models. Such tools should require little or no programming knowledge of the user, have a user-friendly graphical interface, provide user presentation development tools and have a fast simulation engine for rapid modeling of complex systems. In addition, the results of

7

FIG. 3 illustrates a chart generated in accordance with a preferred embodiment of the present invention.

FIG. 4 illustrates a view of the navigator screen in accordance with an embodiment of the present invention.

5   FIG. 5 provides a view of a dashboard screen in accordance with a preferred embodiment of the present invention.

FIG. 6 shows an exemplary table of the icons within the application toolbar of an embodiment of the present invention.

FIG. 7 shows a table of icons for a control toolbar in accordance with an

10   embodiment of the present invention.

FIG. 8 shows an implementation of aspects of the present invention within a computer environment.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

15   Preferred embodiments of the present invention provide a system dynamics modeling environment with a graphical user interface that allows a model developer to define a model through graphical inputs, for example by selecting icons and dragging them onto a representation of a flow diagram. Most preferably, graphically input data and other information regarding a particular model is stored within a data structure that

20   efficiently organizes the data. Such a data structure is preferred for the organization it provides to model development and because the database organizes and expedites the compilation and execution of the model.

Certain preferred implementations of a system dynamics modeling environment allow the definition of a plurality of groups, with each group consisting of a data set representing a complete flow diagram and capable of being coupled to another group defining a portion of the model. Variables within a group may be defined solely within

5   the group or may be shared with one or more other groups. Thus, when a model including at least two groups is compiled, the compiler recognizes the presence of a shared variable within the two groups and links the two groups together. As with conventional system dynamics programs, it is possible in the preferred modeling environment that individual variables on a layer may contain within them one or more layers of sub-functions or

10   models, with each layer defined by a flow diagram and with variables shared or passed between the layers. The group structure provided in preferred embodiments of the present invention is in addition to such a multilayered structure.

The present invention may be embodied as a computer program product including some or all of the preferred methods that facilitate the construction and simulation of

15   large system dynamics models that might include thousands of equations and array elements. A fast, efficient simulation engine enables certain particularly preferred embodiments of the present invention to combine the ease of use of graphical interfaces with the computational power to simulate large, complex system dynamics models. This is believed to provide a significant advantage to the application of system dynamics

20   methodology to increasingly complex systems while also allowing the end products of modeling to be cost-effectively delivered with familiar and easy to use interfaces.

Preferred implementations of the present invention provide a graphical editor to facilitate the entry of data sets, relationships and structures for very large system

12

suited to graphical manipulation, such as overall structural layout, and the text-based

variable definition editor can be used to define aspects of the model most appropriate to a

textual or equation processing interface. A user in the modeling environment preferably

automatically creates the variables in graphical form and displays the variables in a

5   separate flow diagram window. Any variables or functional relationships created

graphically will show up in the text editor. Once the variables and equations are created,

the user has a choice of editing aspects of them and their relationships directly in either

the graphical or text editor.

A distinct aspect of certain preferred embodiments of the invention is a model

10   builder "navigator" interface that is most preferably distinct from and in addition to the

graphical interface with graphical and text editor capabilities. The "navigator" interface

is complementary to the graphical and text editors and provides a model developer with

model elements in a hierarchical, tabular or chart-like display so that the developer can

easily store, locate, and retrieve major model components, such as groups, variables,

15   plots, equations and macros. The navigator feature allows a developer the ability to view

and edit a substantial quantity of data in an organized format. For example, the model

developer might use the navigator to view the relationships among a number of variables

and access the equation variable dialog boxes that define the relationships among the

variables. Such editing can be difficult using a standard, single window graphical

20   interface, and is greatly facilitated using the navigator function. It is particularly

preferred that the system dynamics modeling environment provides a navigator interface

in addition to the text and graphical editing capabilities.

14

Certain particularly preferred embodiments of the present invention provide

advanced compiler and simulator functions that significantly reduce the time it takes to

simulate large models. The use of advanced compiler and simulator functions reduce the

time it takes to develop and test models and allow a greater number of tests and model

5   variations, also known as scenarios, to be considered in a practical manner. Another

preferred aspect, which is in some regards synergistic with the faster compilation and

simulation aspects of some embodiments of the invention, allows the precise

identification of scenarios to allow for many scenarios to be run and accurately tracked.

Scenario management capabilities of certain preferred embodiments of the invention

10   enable the specification, storage, and retrieval of a virtually unlimited number of

alternative model scenarios.

Other preferred aspects of a system dynamics modeling environment in accordance

with the present invention provide a facility for developing user interfaces and a separate

facility for defining informative and flexible charts and graphs. A preferred user interface

15   facility provides a "dashboard builder" that enables users without any programming

experience to create professional-looking graphical user interfaces for their models.

Advanced charting features enable users to create a wide range of two-dimensional and

three-dimensional charts with "drill-down" capability. Such charts are, of course,

significant to explaining and illustrating the results of simulations.

20   A number of other design features are worth noting. Preferably, the system

provides an "open-enough" architecture so that other languages or applications can be

coupled to models or the modeling environment itself to extend the application. For

example, one embodiment of the modeling environment directed to this objective

15

provides a complete implementation of some or all of the aspects of the present invention

as a stand-alone ActiveX component. Other applications of the present invention may be

coupled to or operated in conjunction with programs written in Visual Basic, Visual C++,

Excel or Word Macros, or the Windows (98) Scripting Host. Those of ordinary skill will

5   recognize that the intended outputs and interfaces of the present invention will vary over

time and that this list of functionality is representative only.

In conjunction with or distinct from these "usability" interface functions, the

design of preferred embodiments of the present invention allows models produced by the

modeling environment to be modular so that the environment supports component-based

10   model assembly and distribution. Also in keeping with high quality software design

principles, preferred implementations of the present invention support the clear

documentation of all model components and their evolution. This allows for the ready

modification and distribution of new or replacement component parts of a model as the

model is refined or further developed.

15   The term "model" as used herein refers to several related aspects, including the

equations and the overall project that is the subject of the model. The definitions in the

following list clarify a number of other terms used in the present discussion:

Model   The entire set of information that is used in a modeling project,

including simulation specifications, parameter sets, plot

20   instructions, and dashboard layouts.

SDMDB   A System Dynamics Model Database contains some or all of the

information for a modeling project. It may also contain all or parts

of multiple models, model versions, and model variants. An

SDMDB may be distributed as a single entity.

Model Structure	The equations and associated flow diagrams that define a model.
Model Version	A specific set of model structure equations that is different from other "versions".
Group	A collection of model definition elements including equations and associated flow diagram representations.
Scenario	A scenario is a specific model simulation that is based on a particular version of the model and a particular set of parameter values.
Parameter	An input variable that is used in model calculations and whose value helps determine system behavior.
Parameter Set	A collection of parameters that can be invoked or referenced by parameter set name.
Model Data	Static information, typically actual system behavior, about modeled system behavior which is used as a point of comparison and is generally not involved in calculating model behavior.

Building and simulating system dynamics models within a modeling

environment provided by the present invention preferably involves an iterative process.

First, a model is built by defining individual model elements such as variables and the

relationships among the variables. This may be done using the graphical editor, a textual

editor or, more preferably, both types of editors both sequentially and in cooperation.

17

Simulation of the model proceeds by defining initial variable states and the length of the

simulation time period. The simulation function preferably includes a compilation

function to expedite the processing of the model. At the end of the simulation interval,

the simulation results can be viewed. A number of tests might be performed on the model

5   at this time. For example, it will often be appropriate to review the values of variables

through the system to determine if they are taking meaningless or void values. Also at

this point, it is preferred that the simulation results be compared to model data to evaluate

the accuracy of the model and the parameters provided to the model. Based on the results

of the simulation, the user can then re-specify model variables to modify the results. The

10   process of running a simulation and testing the simulation results against model data is

repeated until an appropriate model definition and parameter set is found.

After a model is completely defined, it can be packaged in a form for distribution

to end users who will execute the model for different sets of input data to examine the or

predict results on the basis of the data sets. The scenario or model is preferably combined

15   with an appropriate set of user interface controls by using the dashboard functionality of

preferred aspects of the present invention. In addition, the scenario or model is preferably

combined with a desired selection of graphical output displays, adapted to the particular

data to be displayed, before the scenario or model is distributed to end users.

20   *System Dynamics Model Databases*

Most preferably, data and other information related to a model are stored within

system dynamics model databases (SDMDB's) within the modeling environment. The

database structure is a fundamental characteristic of the modeling environment and

18

model or create a new model, which in turn preferably selects the associated SDMDB or

SDMDB's to access. In other embodiments, the modeling environment prompts the

developer to identify the initial SDMDB for a session. The developer may open two or

more SDMDB's at one time. Scenarios, groups, or elements may be copied or moved

5   between SDMDB's. Some embodiments of standard SDMDB's are write protected and

may be used only as the source, and not as the destination, of a file transfer.

In some embodiments of the present invention, provision is made so that an

SDMDB may be a library of reusable and independently testable model elements.

Preferably, such a database is read-only to the typical user or to users defined below a

10   certain level of access priority. Thus, a modeling environment may include or have

access to SDMDB's that are read-only and cannot be altered in a typical modeling

session. Preferably, facilities are provided within the modeling environment so that an

SDMDB can be generated and made read-only ("locked") to future users or to users that

receive a distribution version of the SDMDB. For convenience, if the user or developer

15   closes the modeling environment from within a model, the screen is restored to the

condition as of the time of the shutdown. This facilitates resumption of work with

minimal delays, which is well appreciated when working on a complex system.

Groups and System Dynamics Models

20   To aid the model building process, developers can place collections of variables

and other model elements inside model groups. The concept of the group is a particularly

preferred aspect of the present invention that assists the user in more quickly constructing

models, as well as assisting the compiler in more quickly running scenarios. The group

approach is especially useful when constructing large models that consist of many model

sectors, each representing a different business focus or useful output of the model. For

example, a model may contain a financial sector, manufacturing sector, purchasing sector,

inventory sector, etc., and each would be contained within a single group. Unlike system

5   dynamics computer program products in the related art that force the user to place all

variables on one or more layers of a single flow diagram, users of a preferred

embodiment of the modeling environment can place various model sectors into defined

model groups. A model then is a collection of individual but tightly interrelated groups

of variables and other model definition elements. When a model is defined within a

10   plurality of groups, each group preferably has defined within it an interface to one or

more other groups.

By way of illustration, FIG. 1 might define a single immune system group within

a model of disease within the human body. For this embodiment, the screen of FIG. 1

may be a group definition screen and each group within the model would have a

15   corresponding screen that could be called up into the graphical interface by accessing the

group for display and or editing. Each of the groups with their associated screens has a

name unique within a given model. The group can be accessed, for editing or otherwise,

by referencing the name associated with the group in the group definition process

facilitated through the screen of FIG. 1. The group definition screen of FIG. 1 shows

20   what might be a substantially complete model of the interaction between the immune

system and a pathogen. More complicated or simpler models might be contemplated but

are not shown here. Other groups within the larger disease model might include, for

example, a group modeling the circulatory system that interacts with the illustrated

21

immune system group to inform a rate variable for pathogen transport within the immune

system group. The complexity of the immune system group shown in FIG. 1 illustrates

in part why it is difficult to define a large model within a single flow diagram.

Different groups within a larger model are defined so as to communicate peer to

5   peer with other groups. This group definition is distinct from another defined and

conventionally available capability to define models having layers of complexity

incorporate within a give variable or model element.

An embodiment of the present invention provides a computer program product in

a computer-readable media for use in a computer to build complex system dynamics

10   models using a plurality of discrete hierarchical system dynamics groups. Each group is

a set of interactive model definition elements, and the dynamic relationship among the

model definition elements of all groups in the system model defines the structural and

quantitative relationship among the variables that comprise the system model. This

embodiment preferably includes a means responsive to user selection of a plurality of

15   system dynamics groups, a means for hierarchically arranging the groups in a system

dynamics model structure, a means for copying a variable from a first group to a second

group, and a means for defining the dynamic relationship among the groups to describe

the effect of changes within the group on the system model and the effect of system

changes on the group. In a preferred implementation, the computer program product

20   simulates the performance of the complex system dynamics model. This preferred

implementation preferably includes a means for enabling the groups to dynamically

interact during initialization of the system model and as the system model is exercised in

a simulation.

22

Another embodiment of the present invention is a computer program product in a

computer-readable media for use in a computer to build a system dynamics group for use

in constructing a complex system dynamics model. The group, which has a unique name,

includes model definition elements and may have both equation and graphical

5   representations. If the model includes multiple groups, it is preferred that each group

includes at least one interface to another group, including at least one reference to a

variable in another group.

Preferred embodiments may include a means responsive to user selection of a

plurality of model definition elements for a group. A first group might include model

10   definition elements that are selected from the following:

o   Variable type;

o   A second group;

o   A variable from a second group; and

o   A flow diagram.

15   The group can also include a list of SDMDB's and models to search for additional groups

and can further contain simulation specifications.

The variable types that might be defined include:

o   Level;

o   Rate;

20   o   Auxiliary; and

o   Constant.

23

Of course, this list of variables is not specific to an embodiment in which multiple groups are used to define a given model. These are the variables that might be used in any model, regardless of complexity. Groups provide several benefits, such as:

- o Modularization and organization of model definition elements;

5
- o Support of rapid navigation to reach specific functional areas of the model;

- o Support for views of related information;

- o Access to reusable equations;

- o Automatic flow diagram preparation and maintenance;

- o Flow diagram modularization;

10
- o Easy means for the user to change a group definition for one or more alternative structures;

- o Support for distribution of a standard library of Macros;

- o Reusable data structures that may be replicated many times within one or more models; and

15
- o A natural structure for component by component model testing.

Bezier curves are supported for information links and general sketching. These information links and flow links move intelligently when objects are moved. The user may control the shape and path of the curve by using mouse-based drawing tools. In

20 addition to group objects and links, the user may place text (labels, titles, documentation), boxes, lines, and pictures on the flow diagram.

Display options for each object, link, and optional display elements are controlled using a properties box. These properties include shape, color, style, size, font, and visibility. The user may change the default characteristics for new objects, by object type. The user may also change characteristics for any individual element or selected set

5   of elements. The user may customize the view by including or excluding ghosts, parameters, and constants.

Free-form flow diagrams contain any combination of variables, parameters, or groups and may be constructed by copying all or part of group diagrams. To the extent possible, free-flow diagrams are automatically updated when a variable or link is deleted

10   or when a variable or link is added to a group or variable that is already in the diagram. Causal tracing can be started from a free-form diagram.

Using the Modeling Environment

Aspects of the present invention might be practiced for the described embodiment

15   of a multiple group model or might be implemented in a different embodiment. Regardless, implementation of a system dynamics model in accordance with the invention may continue after initial database selection by utilizing means responsive to user selection for defining the hierarchical structural relationship of the model definition elements in a group. In practice, this is preferably accomplished in a graphical editor of a

20   type that is familiar and readily implemented by those of ordinary skill in this art. Referring to FIG. 1, a developer builds a model using the graphics editor 1 by selecting a variable type (level, rate, auxiliary, or constant) from a tool bar 2 and places it on the main flow diagram 3.

A preferred embodiment further includes a means responsive to user selection for defining the quantitative relationship of a model definition element to another model definition element in the group. This might be accomplished in an editing environment that is accessed as part of the graphical editor or it might be accomplished using the text

5   editor. Referring to FIG. 2, once a model variable type is selected and placed on the flow diagram of the graphical editor, the user opens the properties box 50 for that variable. The properties box might have an appearance and functionality like that shown in FIG. 2 and can be used to further define and specify the exact form of the variable. Each variable type has a slightly different properties box as appropriate to how the variable

10   relates to other variables. In most cases, the user specifies an equation 51, or refers to a table or function, describing the relationship between the selected variable and one or more other variables on the flow diagram. The resultant definition is stored within the active SDMDB and is incorporated into the overall system of differential equations that describe the behavior of the system constructed within the modeling environment.

15   Preferred embodiments might further include means for copying a model definition element from a first group to a second group. This might be accomplished using conventional "cut and paste" graphical editing techniques between two group display windows each like that illustrated in FIG. 1. Alternately, this might be accomplished using the navigator functionality described below. In a still other

20   alternative, a form of model definition copying is accomplished by copying an SDMDB and editing within that copy of the SDMDB.

This preferred aspect of the present invention allows models to be constructed much faster than by conventional methods. In a preferred embodiment, the computer

26

program product can simulate the performance of the model consisting of at least one

group. Once all variables have been specified, the user simulates the model by selecting

the simulate command from the menu or toolbar. As will be described below,

segmenting the system dynamics model into groups of related elements enhances the

5   ability of the compiler to run system dynamics programs efficiently and quickly. When a

simulation is initiated, the compiler examines all the variables and equations and

performs checks to ensure that they are properly specified. Once the equation checking is

complete, the simulator solves the equations for each time period until the simulator

reaches the specified end of simulation. Both the simulation time period (or DT) and the

10   length of the simulation are set by the user. For example, a user may specify a simulation

time period of one day, and simulate a model for 365 time periods.

When a simulation run is complete, a developer or user has several options for

viewing results. Simulation output can be viewed in either graphical or tabular form.

The computer product of the present invention preferably possesses advanced charting

15   capabilities that allow the user to view simulation results in a variety of formats. Users

may chose from a range of two-dimensional and three-dimensional bar, column, pie, line

and area charts. An example of one form of graphical output is shown in FIG. 3.

Preferred embodiments of the modeling environment include a means of

generating a graphical user interface (GUI) responsive to user commands input through a

20   graphical editing tool such as a mouse or other implement that interacts with the

graphical editing features of the interface. This GUI can include a means of navigating

through the model to a particular group and displaying a view of the state of the group.

The GUI, aspects of which are illustrated for example in FIGS. 1-4, includes a screen

displaying the hierarchical structure of the model, a means for selecting a group from the hierarchical structure of the model, and a means for presenting a plurality of views of the selected group's attributes.

For creating and editing the flow diagrams and models, the program preferably
5   provides some standard shapes, such as boxes for levels and valves for rates, shown on the left sidebar of the FIG. 1 display. Additional shapes are provided to offer additional flexibility to the user in developing representations as required. Every variable calculated in a group is preferably placed in the group flow diagram. The user drags and drops to position these objects. Inputs from other groups may be optionally displayed as ghosts.
10 Output to other groups may also be optionally displayed. The user may add or delete links using a group flow diagram such as that illustrated in FIG. 1. Further changes as appropriate might then be made through the variable dialog box. Some changes require that the user review and modify the underlying equation or equations and the interface preferably prompts the user or model developer when such a change is made. The
15 equation may be modified in the text editor or in the variable dialog box that can be accessed by clicking on the element in the flow diagram.

Macros and Molecules: Reusable Groups

Groups may also be used as macros or as molecules, which are groups specifically
20 designated as reusable and distributable. Groups designated as macros or molecules contain equation structures that can be used (i) by more than one model, (ii) more than once within a model, or (iii) both.

Macros within the modeling environment of the present invention are fully compatible with existing "DYNAMO" macros. A version of DYNAMO is described in the book by Richardson and Pugh, Introduction to System Dynamics Modeling with Dynamo (Productivity Press) 1981, which is hereby incorporated by reference. Macros

5   are specified by setting the group property for "type of group" to macro. With this specification, the group definition screen lists the calling arguments for the macro in order and indicates whether they are input or output arguments. A macro may contain internal variables not included in the argument list. Macros are invoked in equation mode by using their group name.

10   "Molecules" are extensions of the macro construction. Any thoroughly developed group is suitable for reuse as a molecule. If a group is used as a molecule, it is referred to as a molecule; otherwise, it is simply called a group. Molecules most preferably contain all of the information necessary to make them reusable and commercially distributable. Thus, a molecule contains not only equations and flow diagrams, but also appropriate

15   macros, exogenous data, and parameters. Particularly preferred implementations of the molecule function provide help information that assists the user in using the molecule part of a model.

Molecules connect to the rest of a model by connecting variable names defined within the molecule to variable names defined within the model and the SDMDB for the

20   model. A molecule may be used many times within a model. To function, however, the model must either map every output variable from the molecule to a variable externally defined in the group or use the extended name for all references. A molecule might also be used as an array:

29

VARNAME[RANGE1, RANGE2,...] = MOLECULE.

This usage will essentially make every output variable in a molecule function as an array. The reference statement may then be:

VARNAME [RANGE or ELEMENT] .MVARIABLE.

5   If the molecule contains array elements, each array is preferably assigned one or more additional dimensions:

VARNAME[RANGE or ELEMENT] .MVARIABLE[RANGE or ELEMENT]

Molecule arrays may be used to construct models with predefined model subsets having dimensions varying according to the model or model version. If a model input variable is

10   an array that uses the same range as a molecule equation, the appropriate element of that input array is preferably used for each affected molecule calculation.

Macro and Molecule Libraries

To facilitate the use of macro and molecule functions, the modeling environment

15   allows the user to specify a list of external SDMDBs to be searched to find macros and molecules that are not found in the current model. This definitional function within the modeling environment most preferably also allows a user to define an explicit source for each macro or molecule. To the extent possible, the modeling environment functions effectively the same way in all views of the model. The user may select any group in the

20   navigator or flow diagram by clicking on the group name. A right click opens the Group Actions menu with the selections:

Open flow diagram - Opens the primary flow diagram window for the selected group.

30

Open text editor which opens the text editor view for the group.

Properties Window, describes group properties and other comments.

Insert New Group Before/After, where new groups may be added at any time. The user is able to create a new group within the model definition of the SDMDB by

5   entering its name and position. The user may also select a group from the current model or from any available SDMDB.

Delete Group, which the user may use to "ungroup" the selected group and to assign all of its components to the next higher group.

Replace Version, which the user may use to change the source or version of an

10   included group. Old versions of groups may be included in the current version.

Compiler Functionality

Most preferably, the modeling environment provides the ability to build and execute simulations of system dynamics models and to make the results available for

15   review and analysis. This is facilitated in particularly preferred embodiments by advantageous aspects of a preferred implementation of the environment's simulator. For instance, in a preferred aspect of the present invention, the simulator is an integral part of the modeling environment, not a separate program. When a simulation is requested, the modeling environment preferably recodes the model equations into tables and structures

20   that are used during the simulation execution phase. When appropriate, these structures are maintained over the life of one or more simulations and preferably support multiple executions with minimal overhead. The core of the simulation engine most preferably is implemented in performance-coded assembly language to provide high execution speed.

31

Unlike a traditionally programmed model, a system dynamics model does not

contain a sequenced list of instructions. Users are free to type in equations in any order

they wish and, by the very nature of the graphical model definition process, equations and

variable references will exist within the model (actually, within one or more SDMDB's)

5   in arbitrary order. To generate a simulation, the simulator generally determines a correct

sequence of calculations based on the equations existing within the definition of the

model. This is a process that is conventionally required of system dynamics software

programs in initiating a simulation. Preferred implementations of the present invention,

however, provide improved simulator performance by more efficiently performing this

10   process and, most preferably, producing a structure that is itself more conducive to

efficient further processing.

The integrated design of a simulator in accordance with the present invention also

provides a more flexible simulation environment. "DYNAMO," which is perhaps the

most powerful of the conventional system dynamics modeling programs now available,

15   determines a correct sequence of execution during a "compiler" step that precedes

simulation. The "DYNAMO" compiler does not support the execution of potentially

correct models that involve references within arrays or conditional dependencies.

Simulators in accordance with the present invention, on the other hand, support such

constructions by providing run-time re-sequencing of equations and variables. Although

20   this technique is commonly used in spreadsheet software, it has previously been thought

to provide inadequate performance for large system dynamics models. Particularly

preferred embodiments of the present invention address the difficulties of run-time

sequencing through the use of individual instruction pointers for each element of each

32

array as well as the use of relative addressing within the instruction tables through p-code. The simulator preferably also produces sequenced lists of variables that are to be calculated at a particular time, which can further reduce the overhead of the simulator checking to see if a particular variable has been calculated.

5   Structuring model elements into groups is a particularly advantageous feature of the present invention's modeling environment for enhancing the speed at which simulations can be run. Essentially, by so grouping model elements, certain of the groups can be processed less frequently or even removed from the processing sequence until variables within the group are altered. A system dynamics model typically requires the

10   execution of a number of equations for a specified number of time periods. A large model might include major segments of equations, or groups, that are not executed during significant portions of the simulation period. "DYNAMO," as an example of a system dynamics modeling system capable of processing large models, simulates all of these equations whether the simulations alter the variables within those groupings or not.

15   To reduce simulation times, the compiler of the present invention is designed to identify the sets of equations that preferably are executed during any given simulation run. In this way, particularly preferred embodiments of a simulator reduce simulation execution time by identifying groups of equations that are 'frozen" during detectable intervals, and therefore need not be executed. In these embodiments, the simulator keeps

20   track of which variables are active and which are frozen, thereby ensuring the accuracy of simulation calculations while significantly reducing the execution time. Further to these aspects, the modeling environment preferably maintains a list of variables that are known to be uniquely required by a group, which variables are uniquely produced by a group,

33

◊   The Units, Legend or Comments;

◊   The Saved checkbox; and

◊   Inside a Macro, the Macro Argument checkbox.

○   If any differences are found, a Model Differences window is shown. This

window is a regular application window, not a dialog, allowing it to be moved

and resized.

Generating Scenarios

The scenario can be a useful tool in analyzing model behavior. Typical modeling

analysis involves examination of a number of scenarios. A scenario is a specific set of

parameters and a specific model version that produces a certain set of results. For a

useful model, a scenario should produce the same results each time it is run. Variable

changes that can alter the results constitute a new scenario. A scenario always relates to a

specific version of a model and to a specific set of parameter values. The user may create

a new scenario by changing the model or the parameters values. Conveniently, the

scenario number is assigned in numerical sequence within the SDMDB.

Users have access to features that make it easy to create and save scenarios and

models. Using the scenario specification grid, users can easily retrieve model parameters,

change parameter values, store parameter sets, and execute and save new scenarios.

Model parameters are accessible in a tree, similar to file folder trees conventionally found

in "Windows" applications. Multiple parameters can be selected and changed at once,

and these changes can be grouped into parameter sets that can be saved for later use.

Model simulation runs, *i.e.*, scenarios, that are executed based on these parameter changes

can be named and are automatically saved by the program. Users can also view a

complete listing of saved simulations in the scenario roster. The scenario roster shows all

the saved scenarios, a list of parameters that have been changed, and summary output

results for each.

5   The user may save the results of a simulation run. The "save" command closes

the scenario and freezes all of the inputs required to generate the result. The user may

also explicitly discard the results. If the results of a simulation have not been explicitly

saved or discarded, the user is preferably forced to choose one of the above prior to

exiting or to making changes that would prevent a "save" command from being

10   successful. The specifications, version, and results are saved by default unless the user or

end-user makes an explicit decision not to save the scenario. If the user decides not to

save the results or to freeze a specific set of definitions as a scenario, then the scenario

definition is not saved.

The scenario list is used to select which scenario to use as a base for other

15   scenarios. In browse mode, the user can quickly scan the assumptions that distinguish

among the various scenarios. The list format may also be used to select scenarios to

review. A grid is used to review and select scenarios. The scenario list box may also be

used to make a scenario active, which makes it available for comparative analysis with

the current scenario. If a user wishes to make an already executed scenario the current

20   scenario, any work in process must first be discarded.

Scenario numbers are unique within an SDMDB and are assigned starting with

the number 1. The default name for a scenario consists of the model name and an ordinal

47

number sequentially designating the scenario. Alternate names and aliases may be

supplied by the user. Scenario numbers are not re-used within a given model, even if

certain scenarios are deleted.

The user may change any parameter used by the model. The user may construct

5   organized sets of parameters to simplify the search for a desired parameter for change. A

list of parameters that have been changed during a scenario is always available. A similar

list is also available for any past scenario. Parameter sets may be defined independent of

a particular scenario. These sets are intended for use when a single group of parameter

values is run several times in conjunction either with other groups of parameter values or

10   with several model variations. Parameter sets are assigned names that identify a

particular list of parameters. The user may create multiple versions of a parameter set

and specify that a model be executed using each set of values.

Parameter sets are created at the model level. In other words, the parameter set

name is unique within an entire SDMDB. Each parameter set contains a set of

15   parameters and values for those parameters. Parameter set membership, meaning the list

of parameter names, may not be changed once the parameter set has been used for a

specific scenario. This restriction may be removed if the detail for each parameter is

stored for each scenario. Every time a value is changed in a parameter set, a different

version of that parameter set is preferably stored under this variation. The version

20   number is simply the number of the scenario that first used that set of values. A specific

parameter set is saved for each scenario. This parameter set contains the names of any

explicitly used parameter sets, and their versions, as well as any parameter values that

were changed for the scenario but that were not explicitly part of a named parameter set.

48

The results of each scenario are stored in a results area that contains variables and time periods designated for retention. Most interactive results analysis consists of either examining the results of one scenario or of comparing values between several scenarios. By default, the most recently executed scenario is available for review. The

5   user may change the scenario being analyzed and build lists of scenarios to be analyzed.

Output Charts for Scenarios

Many of the output charts and tables include the results from more than one scenario. For analysis purposes, the user may select any scenario as the current scenario.

10   Any other scenario may be designated as the base scenario. In addition, other scenarios may be designated as active scenarios. The impact of these selections will vary according to the output display method selected.

The user may select a variable to display by clicking on it anywhere within the model structure (navigator, flow diagram, or text editor). By right clicking within the

15   model structure, several options appear including the display of the default plot.

The program includes a number of standard plot capabilities, including line plots, scatter diagrams, pie charts, and bar charts, both vertical and horizontal. A default plot method is always available for graphical display of any variable. The initial default type is a line chart that shows the values of that variable for the current scenario compared to

20   values for any other scenarios that may exist and that have been activated for this purpose. If the user wishes to see an alternate presentation of the information, a right-click within the plot window will allow the user to change to an alternate format or to a

customized chart. The user may also change the definition of the "default" plot. The plot

window remains active until the user explicitly closes it.

Thus, the user may have many plots on the screen at one time. The user may

further specify that all actions are to be interpreted as analysis. In such a case, the left-

5   click of the mouse on any variable causes the default plot to be displayed immediately.

Tabular output may be viewed in much the same way as charts. The tables may

be selected from standard comparative displays or from custom tables. The user can

specify chart or tabular output by using the "define output" features. Each plot produces

an output window that can be configured, sized, and placed using a standard Windows

10   interface. The user may also specify algebraic expressions whose results are displayed.

Once a plot, chart, or tabular report has been configured, the user may elect to

save the definition for that report. The specification may include the option to use the

variable current as of the run-time or to use the original variable value set at the time the

model was built. Once a custom plot is defined, it is available through a list box when

15   the user right clicks on an appropriate item. These plot definitions may be associated

with a specific group and are displayed in the navigator tree. The plot specifications

preferably include:

　　　　o   Line chart specification;

　　　　o   Bar chart specification;

20   　　o   Pie chart specification;

　　　　o   Scatter diagram; and

　　　　o   Table output specification.

Users also have the option of saving groups of plots, or "plot sets." A single plot

set that contains multiple plots can be created, saved and retrieved by the user. The

"click-to-trace" command combines aspects of the click-to-plot, rapid mode, and tree

diagrams. Click-to-trace allows the user to trace chains of cause-and-effect, or drivers,

5   very quickly. With click-to-trace mode selected, every click on a variable will result in a

plot of that variable and its inputs (the drivers). Clicking on one of the drivers will result

in a further trace. With click-to-trace mode set, the user may also view a text box that

displays the corresponding equations. This box is cumulative; it includes the starting

point and any number of levels.

10   Any chart or report may be moved to "POWERPOINT" or "WORD" or many

other Windows products by either copying and pasting or by the drag-and-drop

technique. Scenario results may be directly output in spreadsheet form, for instance into

a format compatible with "EXCEL". These results will use each saved variable as a row

and each "save per" as a column. Output properties may be used to control labels and

15   formats.

Variable Dialog Boxes

Equation Syntax: The equations specified for levels, flows, and auxiliary variables

use the following syntax:

20   o   The calculated variable (element, array, or vector) is identified in the name

box.

o   Time references are not included.

51

o   White space and parentheses may be used to the definer's content.

Level: The variable definition box for a level provides a very structured format for

changes in level. Most preferably, every input and output is required to be assigned a

name. Flow equations may be modified in this window. Alternately, the user can double

5   click to move to a detailed flow screen. Changes in the flow screen automatically update

the equation view in this window. Levels are specified with in and out flows, by name.

Model variables are created for each flow referenced, unless the variable already exists.

If the level is an array, the initial value specification may switch to another screen. The

variables used by section will provide an initial view into model structure.

10   Flow: Flow equations may be associated with specific source and destination

levels.

Tables: Table variables may be specified using the graphical table changer or they

may be displayed in a matrix format.

Variable subscripts will use a graphical tree control for specifying and displaying

15   subscript families (i.e., arrays) and elements to which the subscripts are applied. When

the user wishes to apply a subscript, the user selects the subscript using the variable

dialog box extension pad. Once applied, the family display will show the subscript in the

graphical tree box for the family. It will be assumed that both sides of the equation have

this subscript. If the user wishes to specify a different equation for one or more elements

20   of the subscript families that apply to the variable, the user selects a new equation (using

the index control), selects which elements the new equation will apply to using the

graphical tree control, and enter the equation to be applied. Numerous equations may be

specified in this fashion. If the user wishes to specify equations that have different

subscripts on the left and right side of the equation, the explicit method must be used.

Automated Simulation Control

5   Automated simulation control refers to various analysis activities that consist of

running a number of simulations and analyzing the results. Each of these analysis

sessions begins with a definition of the simulations to run and of the reports or outputs to

produce. Special analysis calculations may be specified for each run and for comparisons

between runs. The setup information for each automated session is saved for future

10   review and for possible reuse. The major types of automated sessions are:

- o   Sensitivity Analysis (also referred to as "grid search");

- o   Goal Seeking / Optimization;

- o   Monte Carlo (Stochastic);

- o   Driver and Potentiator Identification; and

15   o   Confidence Analysis.

Sensitivity Analysis

Sensitivity analysis provides the ability to execute the model a number of times in

order to evaluate the impact of changing one or more decisions or assumptions.

20   Sensitivity analysis begins with a dialog box that prompts for definition of what actions

to perform. The initial screen includes the model version(s) to use and the input

variable(s) to be applied. Each selected version or variable is applied independently of

other versions or variables. Therefore, if two model versions are selected and run against

53

variables with three alternate values, the automated simulation controller will run six

simulations. The user may specify up to eight variables and up to ten values per variable.

For each variable selected for sensitivity analysis, the user may specify a starting

value, ending value, and increment. The user may also specify values or percent changes.

5   For table functions or for time series data, the user may specify values or time series data.

In each case, the user may enter a description for each selection. Results of the sensitivity

runs may be compared by displaying the values for one or more outputs presented with

the assumptions and decisions that caused those results. These tables may be sorted in

ascending or descending order by results. A number of standard graphs are available to

10   provide a visual comparison of the sensitivity scenarios. Any saved variable may be

selected for BEST-WORST-MEAN-MEDIAN output. This type of analysis may also be

used between any "slice" of the output.

Goal Seeking and Optimization

15   Automated goal seeking and optimization (maximize or minimize) functions are

supported by means of a multiple iteration controller. The analyst is able to select an

output variable and to designate one or more input variables or parameters that will be

changed methodically in order to reach a solution, or at least an approximation. The

analyst may specify ranges for each input, and acceptable ranges for additional output

20   variables *(i.e.,* "Maximize year profits but maintain existing staff levels"). The setup

screen for this type of iterative processing allows the user to limit the total number of

iterations or the total amount of clock time.

54

The best results are saved during the process. Following each iteration, the best

fit iteration display is updated. A running count of total attempts is also displayed, as is

the average run time. A variety of algorithms are provided that allows for optional

"pruning" or identification of optimal results.

5

Monte Carlo Simulation

Monte Carlo simulation allows the user to specify one or more inputs that vary

"randomly" according to an assumed probability distribution. The program preferably

supports several commonly used distributions. The model can then be run many times to

10   assess the distribution of results given the assumed distribution of inputs. This can

provide useful information for risk assessment. In order to use Monte Carlo simulation,

the user preferably specifies the input variable(s) to use, what distributions to apply, and

which output variables to track. A detailed log of the results of each simulation may be

reviewed in tabular form. Selected output variables may be viewed with mean, median,

15   maximum, minimum, standard deviation, and other statistical measures.

Driver and Potentiator Identification

Embodiments of the present invention provide algorithms that seek to identify

important non-obvious relationships. For any given variable, the inputs that generate

20   changes in behavior are identified and ranked. "Potentiators" are defined as multiple

factors that, when changed in conjunction with each other, generate a much larger impact

than the same level of changes in individual parameters. The program provides the

ability to search for potentiators among the input parameters. The user specifies the

output variable that is being tracked, and the program will automatically execute the

model many times with variations of parameter values. The output of this search is a

table listing of the changes in output and corresponding inputs.

5   *System Architecture*

FIG. 8 illustrates a further aspect of the present invention in which either the

modeling environment or a model, whether in development or in a finished form after

distribution, may be provided within a computer system 200. The illustrated computer

system 200 includes a main system 202 including a first long term memory 204 that

10   might include a hard disk drive, optical storage, solid state storage or any practical

memory for storing the programming environment and various versions of models and

other data sets related to the use of the environment or models. The main system includes

a processor 206, which in some embodiments is a single processor and in other

embodiments may be an array of processors configured to process models, equations and

15   other model elements in parallel. It should be appreciated that the illustrated embodiment

of the main system may not be a single computer system but might also be a number of

computer systems distributed over a network or in other fashion.

The main system includes a second memory 210, which functions as a

workspace either for the modeling environment or a model such as when a distributed

20   model is being simulated by an end user. The second memory may include a RAM or

other solid state storage workspace and portions of hard disks, as appropriate to include

the proper amount, speed and accessibility of memory for the environment, programs and

models. During operation of the modeling environment, some or all of an SDMDB 212

is preferably stored in the second memory as the active SDMDB for the modeling

session. Shown in the SDMDB 212 are two groups that are defined within the SDMDB,

one of which might be the simulation definition group. As is preferred in many

embodiments of the present invention, the simulator 214 and integrated compiler 216 are

5   present in or accessible from the second memory 210 because these program elements are

integrated parts of the modeling environment.

Attached to the main system 202 is at least one display 220 that might be used in

either developing a model or in running a simulation. Depending on the particular

functions accessed, an appropriate one of the interfaces 222 discussed above is provided

10   on the display 220. Of course, the interface elements 222 shown on the display 220 may

only be aspects of the overall interface, some of which may include functionality that is

defined, stored and operated within one or more portions of the main system 202. Also

attached to the main system 202 is an I/O facility that may include a wide range of input

devices, such as a mouse or other pointing device, a keyboard, a network connection and

15   one or more external data storage devices, and a wide range of output devices, including a

colored printer appropriate to generating hardcopy output from the graphical output

facilities of the modeling environment.

Although the present invention has been described in detail with reference only

to the presently-preferred embodiments, those of ordinary skill will appreciate that

20   various modifications can be made without departing from the invention. Accordingly,

the invention is defined by the following claims.

What is claimed:

1.   A computer program product in a computer-readable media for use in a computer

to build a system dynamics group within a complex system dynamics model, the group

including model definition elements, the computer program product comprising:

5   means responsive to user selection of a plurality of model definition elements for

the group;

        means responsive to user selection for defining the hierarchical structural

relationship of the model definition elements in the group;

        means responsive to user selection for defining the quantitative relationship of a

10   model definition element to another model definition element in the group; and

        means for copying a model definition element from a first group to a second

group.

2.   The computer program product of claim 1, wherein the group is a first group and

15   the model definition elements are selected from the group consisting of: variable type, a

second group, variable from the second group, parameters, exogenous variables, plots,

dashboards, and flow diagram.

3.   The computer program product of claim 2, wherein the variable type is selected

20   from the group consisting of: level, rate, auxiliary and constant.

4.   The computer program product of claim 1, wherein the program product simulates

the performance of the group, the computer program product further comprising means

for enabling the model definition elements of the groups to dynamically interact during

initialization of the group and as the group is exercised in a simulation.

5

5.   A computer program product in a computer-readable media for use in a computer

to build a complex system dynamics model using a plurality of discrete, hierarchical

system dynamics groups, each group a set of interactive model definition elements,

wherein the dynamic relationship among the model definition elements of all groups in

10   the system model defines the structural and quantitative relationship among the variables

that comprise the system model, the computer program product comprising:

        means responsive to user selection of a plurality of system dynamics groups;

        means for hierarchically arranging the groups in a system dynamics model

structure group; and

15   means for copying a variable and its properties from a first group to a second

means for defining the dynamic relationship among the groups to describe the effect of

changes within the group on the system model and the effect of system changes on the

group.

20   6.   The computer program product of claim 5, wherein the program product simulates

the performance of the complex system dynamics model, the computer program product

further comprising means for enabling the groups to dynamically interact during

initialization of the system model and as the system model is exercised in a simulation.

7.   The computer program product of claim 5, further comprising means for

generating a graphical user interface responsive to at least one user command.

5   8.   The computer program product of claim 7, further comprising means of

navigating through the model to a particular group and displaying synchronized views of

the state of the group, wherein the means for navigating include means for displaying the

state of the group in response to commands received from the graphical user interface

means; and the graphical user interface means comprises:

10   a user viewing screen displaying a hierarchical structure of the model;

            means for selecting a group from the hierarchical structure of the model;

            viewing tracing drivers and "where used" drivers, and

            means for presenting a plurality of views of the selected group's attributes.

15   9.   The computer program product of claim 8, wherein the group attribute views are

selected from the group consisting of: variables, diagrams, dates and outputs.

10.   The computer program product of claim 8, further comprising a means for

selecting a model definition element and displaying all of the occurrences of the element

20   in the model.

11.   The computer program product of claim 7, further comprising a means of

designing a customized graphical user interface for simulating a model in accordance

with a first predetermined set of model definition elements, the set of model definition

elements having at least two variables that dynamically interact.

5

12.   The computer program product of claim 11, wherein the means of designing the

customized interface further comprises

       means responsive to user selection of customized simulator controls and for

displaying them;

10   means responsive to user selection of functional linkages of each simulator

control to a model definition element; and

       means responsive to user entry of an initial state of each variable in the model

definition element and for displaying a result of the custom simulation.

15   13.   The computer program product of claim 11, wherein the customized simulator is a

first customized simulator and the set of model definition elements is a first set of model

definition elements, the program product further comprising means of designing a second

customized interface for simulating a model in accordance with a second predetermined

set of model definition elements, the second set of model definition elements having at

20   least two variables that dynamically interact, where the model definition elements of the

second customized interface include a variable that is not present in the model definition

elements of the first customized interface.

14.   The computer program product of claim 6, wherein the dynamic result of the
interaction of the groups can be stored on and retrieved from a readable writeable storage
device.

5   15.   The computer program product of claim 6, wherein the complex system dynamics
model can be stored on and retrieved from a readable and writeable storage device.

16.   A method of building a complex system dynamics model using a plurality of
discrete, hierarchical system dynamics groups, each group having a set of dynamically
10   interactive model definition elements, wherein the dynamic relationship among the model
definition elements of all groups in the system model defines the structural and
quantitative relationship among the variables that comprise the system model, the method
comprising:

      configuring a system dynamic structure to be modeled by selecting a set of system
15   dynamics groups;

      hierarchically arranging the groups in a structure;

      copying a variable from a first group to a second group; and

      defining the dynamic relationship among the groups to describe the effect of
changes within the group on the system model and the effect of system changes on the
20   group.

17. The method of claim 16, wherein the step of selecting includes assigning a first group a name and assigning a second group a name, and wherein a graphical first flow diagram display shows the first group and does not show the second group.

5   18. A method for identifying model potentiators and defeators defined as multiple factors that, when changed in conjunction with each other, have a comparatively larger impact than the same level of changes in individual ones of the multiple factors, the method including steps of varying the multiple factors.

10   19. A system dynamics modeling tool, including:

a graphical interface for inputting a flow diagram representation of one or more system dynamics model elements;

a definition facility, linked to the graphical interface and accessible from the flow diagram representation, for defining one or more characteristics of a model element; and

15   a navigator interface, distinct from the graphical interface, adapted to display a first plurality of model elements in at least one view of the interface and allowing editing of one or more characteristics of each of the first plurality of model elements.

20. The tool of claim 19, wherein the definition facility is accessed from the graphical

20   interface using a pointing device in conjunction with a displayed flow diagram.

21. The tool of claim 19, further comprising an end user interface development facility capable of providing graphically operable tools for adjusting a value of at least

one parameter and for selecting a type of display illustrating time variations in at least

one model element.

22.   The tool of claim 19, further comprising an end user interface development

facility capable of providing an end user display illustrating time variations in at least one

model element, the end user display automatically provided with a drill down function

allowing an end user to display an underlying definition of the model element by

selecting an aspect of the end user display.

23.   The tool of claim 19, wherein the graphical interface further comprises a group

definition screen, the graphical interface accepting input through the group definition

screen to define a first group and assign the first group a name, the graphical interface

accepting input through the group definition screen to define a second group and assign

the second group a name.

24.   The tool of claim 19, wherein the graphical interface is adapted to accept input to

define a first group and assign the first group a name, and to define a second group and

assign the second group a name.

25.   The tool of claim 24, wherein the graphical interface is adapted so that a flow

diagram display of the first group shows an input from the second group as an indication

of a common variable shared between the first and the second group.

26.   The tool of claim 25, wherein the input is represented in the flow diagram in a manner visibly different from like relationships entirely within the first group.

27.   The tool of claim 19, wherein the tool is stored on computer readable media.

5

28.   The tool of claim 19, wherein the first plurality of model elements and the one or more characteristics of the first plurality of model elements are stored in a system dynamics model database.

10   29.   The tool of claim 19, wherein the first plurality of model elements and the one or more characteristics of the first plurality of model elements represents a first group and wherein the navigator interface is adapted to display characteristics of a second plurality of model elements for a second group in at least a second view of the navigator interface.

15   30.   The tool of claim 29, wherein the first group and the second group are identified by distinct flow diagrams in the graphical interface.

31.   A system dynamics modeling tool, including:

a graphical interface for inputting a flow diagram representation of one or more

system dynamics model elements; and

a definition facility, linked to the graphical interface and accessible from the flow

5   diagram representation, for defining one or more characteristics of a model element,

wherein the graphical interface is adapted to accept input to define a first group

and assign the first group a name, and to define a second group and assign the second

group a name,

wherein the graphical interface is adapted so that a flow diagram display of the

10   first group shows an input from the second group as an indication of a common variable

shared between the first and the second group, and

wherein the input is represented in the flow diagram in a manner visibly different

from like relationships entirely within the first group.

35.   A system dynamics modeling tool, including:

a graphical interface for inputting a definition of a system dynamics model, the

interface adapted to receive variable definitions and equation definitions that define the

model; and

5   a compiler receiving as input a plurality of variables and a plurality of equations

within the model and operating on the variables and the equations to ensure that a

definition exists for each of the variables and equations in the model, the compiler

generating an error message if a variable or equation is undefined.

10   36.   The tool of claim 35, further comprising a simulator to solve equations at each of

a plurality of defined time intervals.

37.   The tool of claim 36, wherein the simulator solves selected equations at fewer

than the plurality of defined time intervals.

15

38.   The tool of claim 36, wherein a time interval and a length of simulation are stored

in a memory and wherein the simulator retrieves the time interval and the length of the

simulation from memory and simulates the model at the plurality of defined intervals for

the length of the simulation.

20

39.   The tool of claim 35, wherein the compiler examines the variables to identify an

occurrence of a duplicate variable definition and to generate an error message in response

to the occurrence of a duplicate variable definition.

68

40.   The tool of claim 39, further comprising a variable reordering facility receiving as

input a list of equations operating on variables and reordering at least one of the listed

equations to cause a first equation that is solved to generate a value of a variable to be

5   executed before a second equation that uses the variable in a calculation, where the first

equation followed the second equation in the list.

41.   The tool of claim 40, wherein the graphical interface generates a flow diagram

representation of the model, the graphical interface further comprising a definition facility

10   accessible from the flow diagram representation, the definition facility adapted to receive

at least some of the variable definitions and equation definitions.

42.   The tool of claim 35, wherein the compiler searches for a functional relationship

between a first and second variable defined as a table function mapping first variable

15   values onto second variable values and, when such a table function is found, the compiler

generates a segmented linear approximation of the table function relating the first values

to the second values.

43.   The tool of claim 42, further comprising a simulator to solve equations at each of

20   a plurality of time intervals, the simulator adapted so that, when a segmented linear

approximation is present, the simulator calculates the segmented linear approximation at

one or more time intervals.

44.   The tool of claim 42, wherein the compiler generates segmented linear

approximations by cubic spline interpolation.

45.   A system dynamics modeling tool, including:

5   an interface for inputting a representation of one or more system dynamics model

elements;

           a definition facility, linked to the interface, for defining one or more

characteristics of the model elements, the definition facility adapted to receive variable

definitions and equation definitions relating variables to other variables;

10   a compiler receiving as input a plurality of variables determined by operation of a

corresponding plurality of equations, the compiler operating to identify a first subset of

the plurality of variables that do not vary in a given time and to identify a corresponding

first subset of equations

           a simulator to solve equations at each of a plurality of defined time intervals over

15   a length of a simulation, the simulator calculating a second subset of equations at each of

the plurality of time intervals and calculating the first subset of equations at fewer than

the plurality of time intervals.

46.   The tool of claim 45, wherein the graphical interface is adapted to accept input to

20   define a first group and assign the first group a name, and to define a second group and

assign the second group a name, and wherein the graphical interface is adapted to show a

first flow diagram display for the first group and to show a second flow diagram display

for the second group.

70

47.   The tool of claim 46, wherein the first subset of equations are the equations of the first group and the second subset of equations are the equations of the second group.

FIG. 1

FIG. 2

3/8

FIG. 3

4/8

FIG.4

FIG.4

5/8
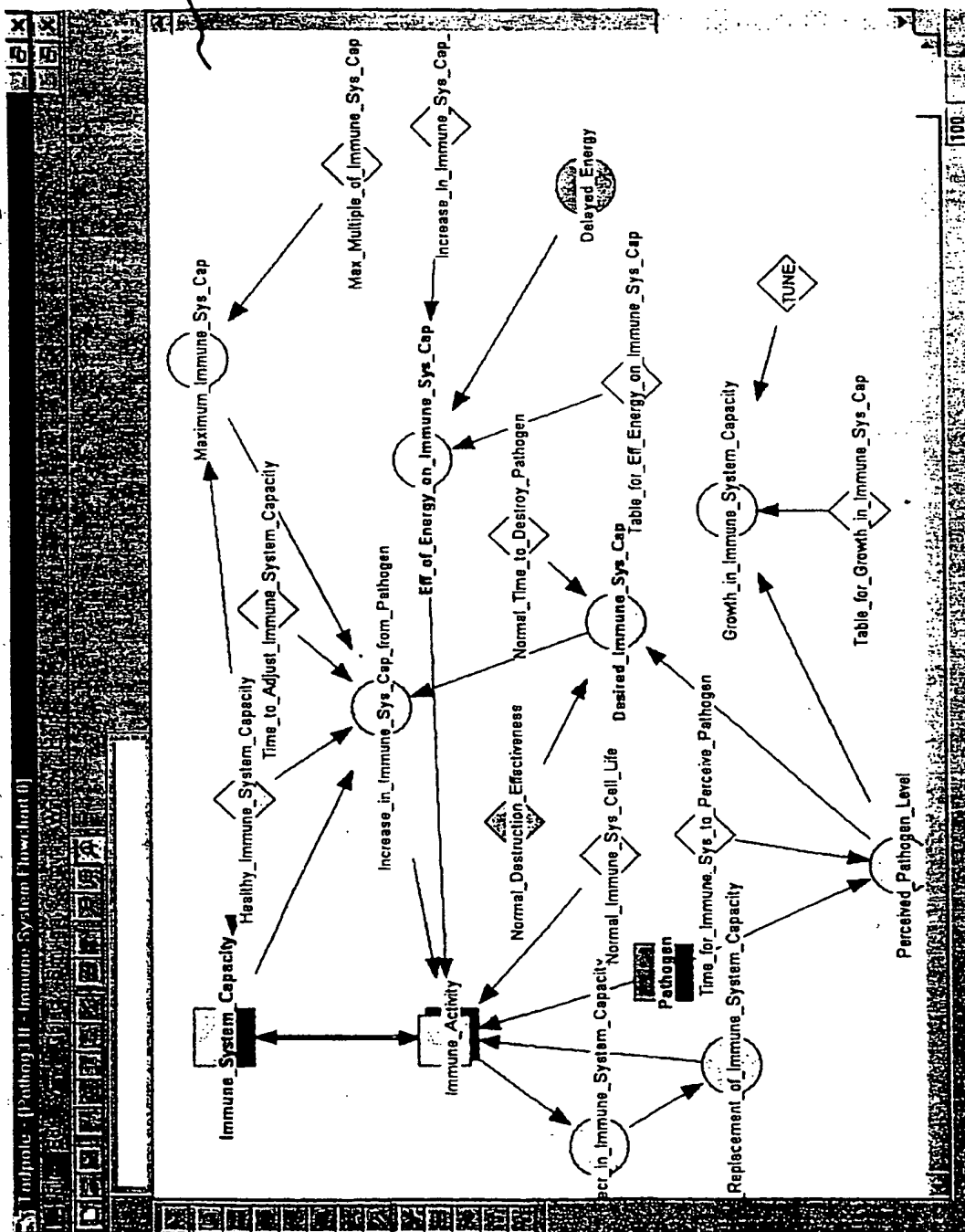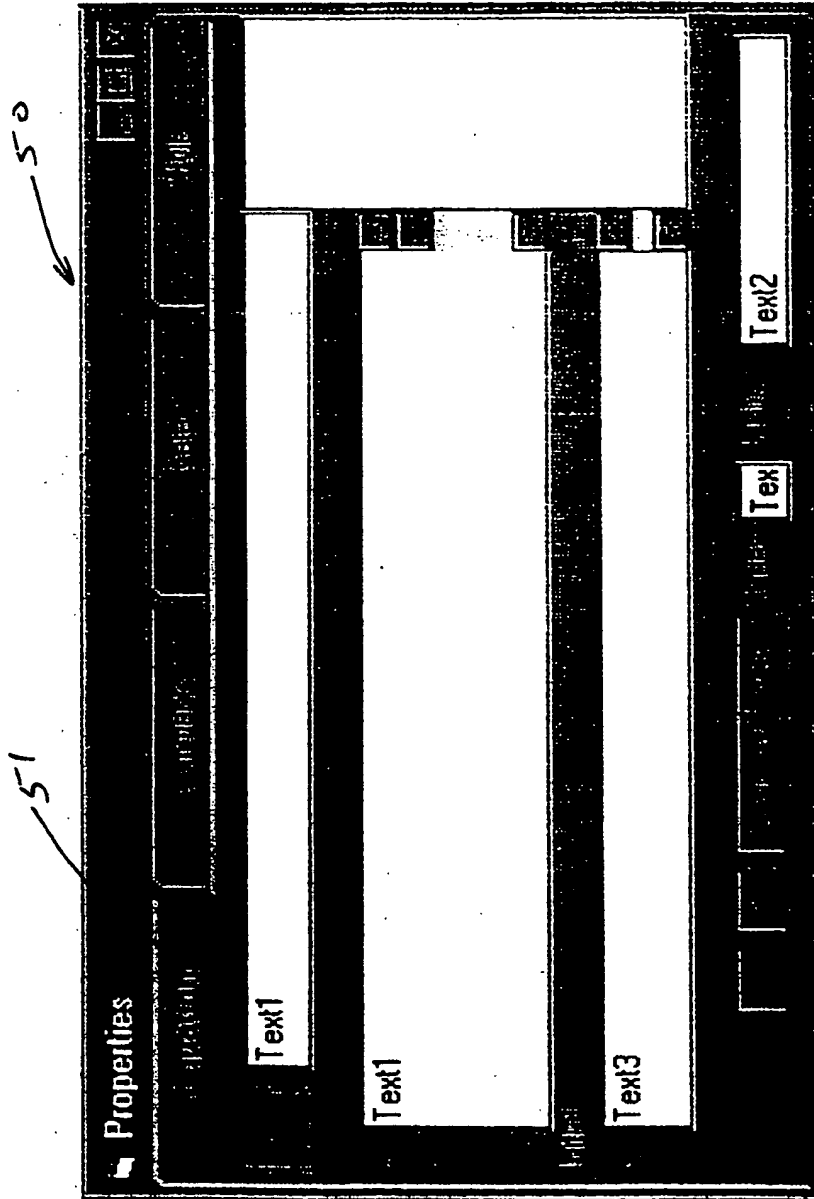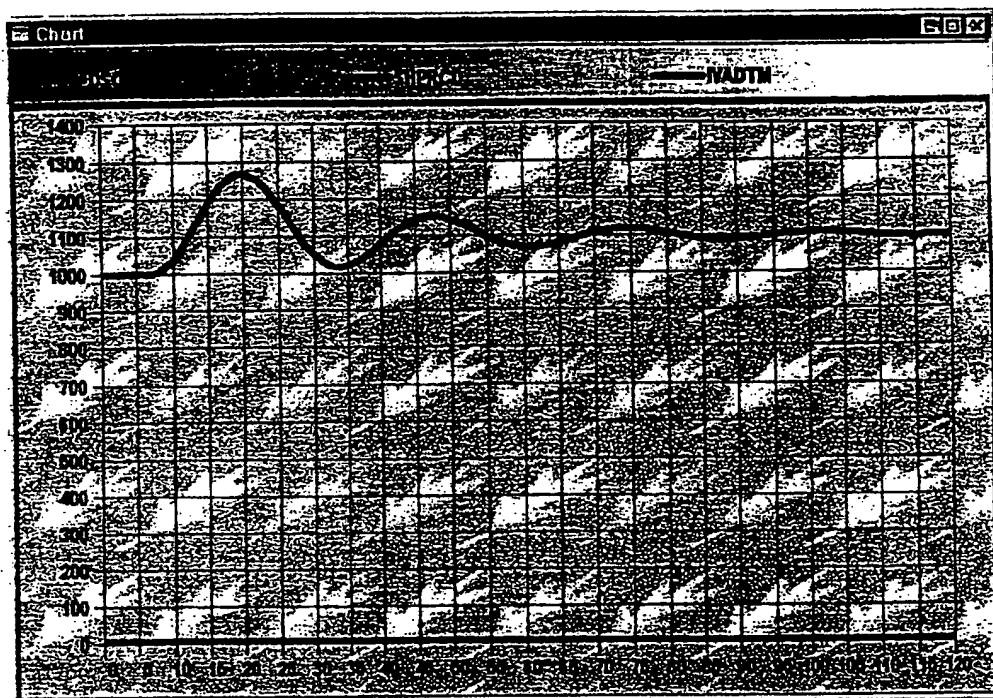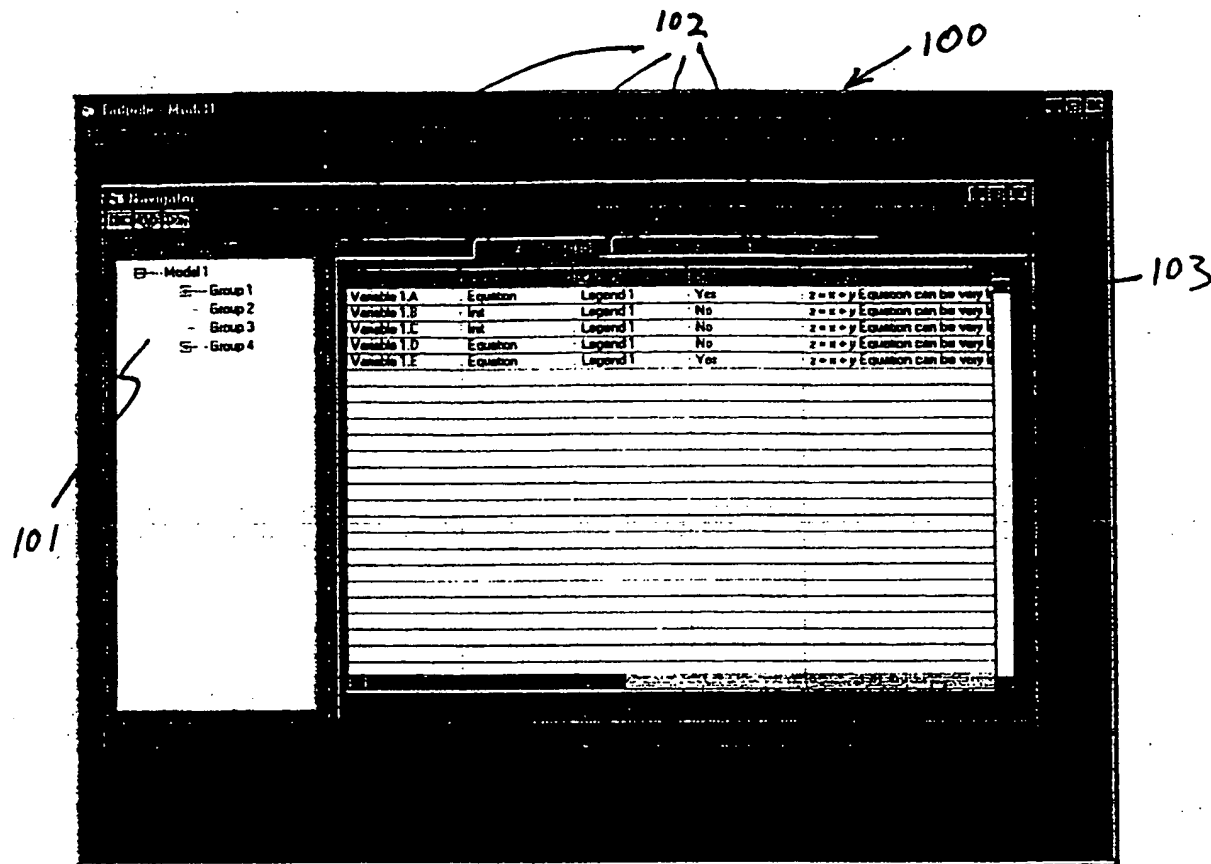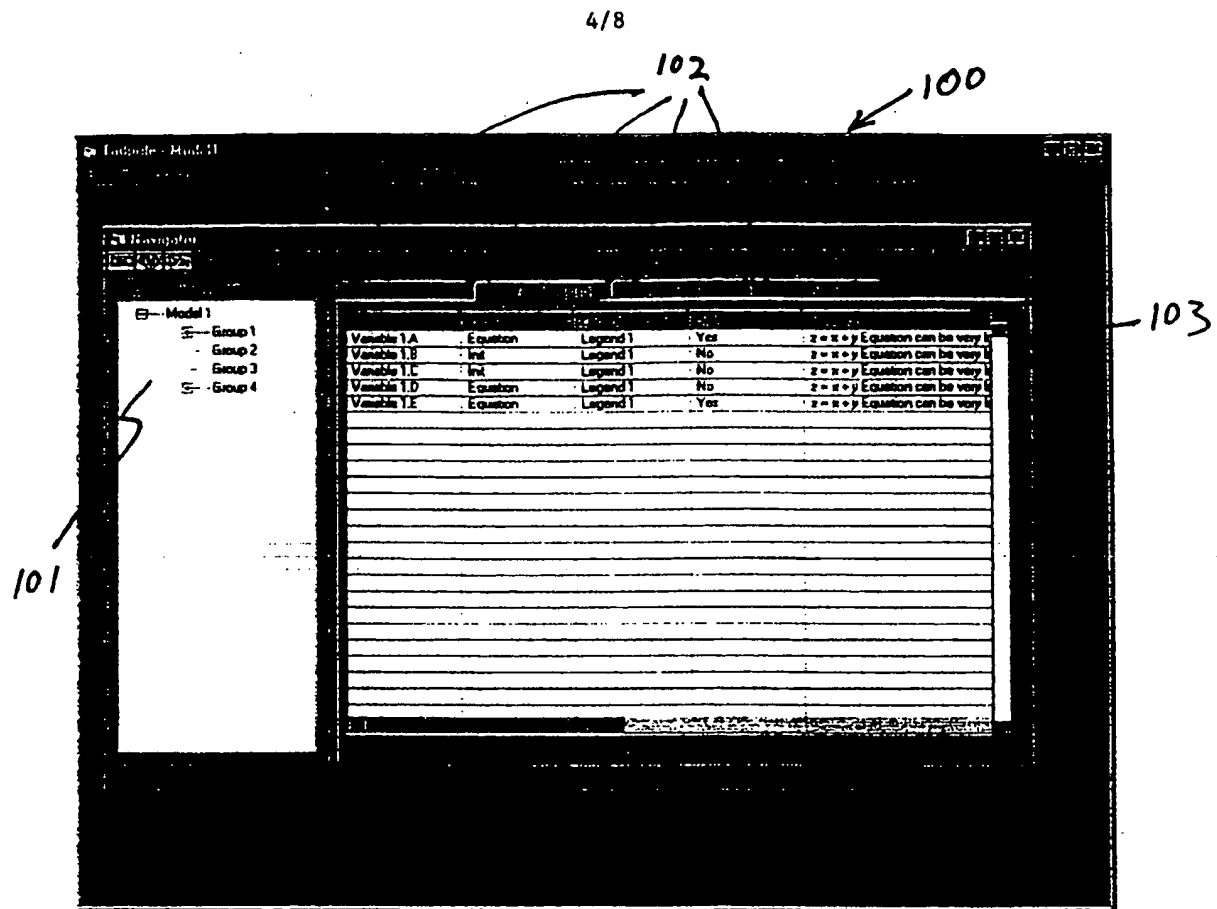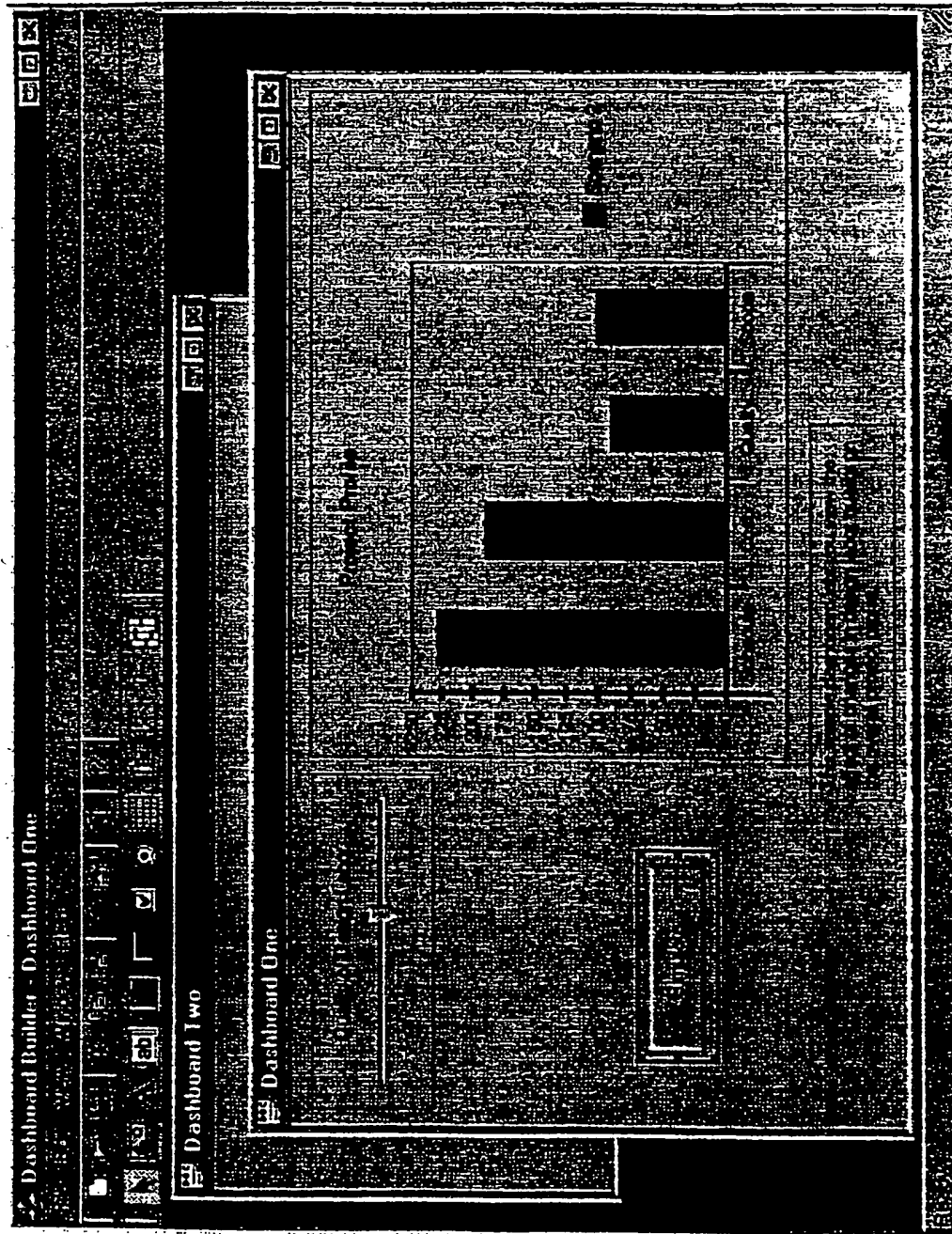
FIG. 5

| Icon	Action
◻	Create a new dashboard document
📂	Open a dashboard document
💾	Save the active dashboard document
✂	Cut
📋	Copy
📋	Paste
◆	Show properties for selected control
◉	Show properties for active dashboard
📑	Show "Insert Control" Dialog
📑	Show "Insert Object" Dialog
▯	Toggles between design and test modes
❓	Show Help

FIG. 6

Icon	Action
▶	Select or Resize an existing control
	Drag out a new Graphic Control
A	Drag out a new Static Text Control
	Drag out a new Edit Text Control
	Drag out a new Group Box
	Drag out a new Push Button
☑	Drag out a new Check Box
⊙	Drag out a new Radio Button
	Drag out a new DB Table Grid
	Drag out a new Navigator Tree
	Drag out a new GTC
○—	Drag out a new Slider
	Drag out a new Gantt

FIG. 7